Integration of Physical Devices into Game-based Virtual Reality

http://dx.doi.org/10.3991/ijoe.v9i5.2705

Z. Zhang, M. Zhang, S. Tumkor, Y. Chang, S. K. Esche, and C. Chassapis
Stevens Institute of Technology, Hoboken, NJ 07030, USA

***Abstract* — Virtual reality (VR) systems have the potential for alleviating the existing constraints on various natural and social resources. Currently, real-time applications of VR systems are hampered by the tediousness of creating virtual environments. Furthermore, today's VR systems only stimulate the human senses of vision, hearing – and to some extent touch – which prevents the system users from feeling fully immersed in the virtual environment. By integrating real physical devices with virtual environments, the user interactions with such systems can be improved and advanced technologies such as the MS Kinect system could be used to augment the environments themselves. While existing development platforms for VR systems are expensive, game engines provide a more efficient method for integrating VR with physical devices.**

In this paper, an efficient approach for integrating virtual environments and physical devices is presented. This approach employs modifications of games that are based on commercially available game engines for implementing the virtual environments in conjunction with the application of Dynamic Link Libraries (DLLs) for realizing versatile communications between these virtual environments and various application platforms, which in turn can interact with the physical devices outside of the virtual environments. This paper is divided into four sections. In the first section, the motivation for the developments described here is discussed, followed by a description of the method used to integrate virtual environments with physical devices in the second section. In the third section, an interactive and collaborative laboratory environment based on a multi-player computer game engine that is linked to physical experimental setups is presented as an example of a VR system. In the final section, some additional promising applications of the developed platform and the corresponding challenges are briefly introduced.

***Index Terms* — DLL, game, virtual laboratory, virtual reality.**

I. INTRODUCTION

A. Virtual Reality Systems

VR systems are representations of actual physical systems with the potential for alleviating the existing constraints on various natural and social resources. By creating and interacting with virtual systems instead of physical ones, it is no longer necessary to consume scarce natural resources to create multiple copies of physical devices. In addition, virtual representations of physical systems can be shared remotely by multiple users and are inherently safer and less failure prone than their physical equivalents. Also, VR systems can be used to implement training environments that do not involve human trainers, do not require co-location of the participants and allow dangerous situations and hazardous environments to be simulated safely (e.g. military training, disaster relief training, firefighter training, etc.). Moreover, recent technological developments have made VR systems accessible from mobile devices, thus enhancing their flexibility.

Although there is no universally accepted definition for it, VR is often considered to be simulations of real or artificial environments based on 2D/3D graphics. The most distinct characteristics of VR implementations are that they are designed to provide the users with a sense of immersion into the environment and facilitate (possibly remote) interactions between multiple users as well as between the users and objects within the environment. At present, most VR implementations focus on the generation of vision and sound perceptions, which are two of the five human senses [1,2,3]. There are four basic types of VR [1,4]:

- Desktop VR [1], which is often used in CAD/CAM [5] and education [6].
- Immersive VR, which often uses head-mounted displays, fiber-optic-wired gloves, position-tracking devices or other devices to immerse the users in the virtual world [1].
- Distributed VR, which is built on several computers instead of one, whereby the users can interact with each other through a network in real time [1,7].
- Augmented VR [1,8,9], which is a form of enhanced immersive VR that can mix real and virtual features to improve the users' feeling of presence.

B. Motivation for Integrating VR and Physical Devices

Obviously, the more vivid VR is, the more meaningful it becomes. However, even though in most instances VR is designed to simulate the real world in a realistic fashion, it still has some shortcomings to be overcome. First, while VR eliminates the physical distances between the users and/or objects in the virtual environment [1], the users' real identities are lost, thus decreasing the credibility of the users and the VR itself. Furthermore, VR is capable of coordinating the users' interactions both spatially and temporally [1] but fails to induce the feeling of presence. Note that the essence of the above shortcomings of VR follows from the word 'virtual'. Despite the fact that one of the main objectives of VR is to mimic the real world as realistically as possible [10], it is impossible – at least today – to make the users feel totally immersed in VR

without actual physical devices. Therefore, the integration of real devices into virtual environments is desirable.

II. GAME-BASED VIRTUAL REALITY AND INTEGRATION WITH PHYSICAL WORLD

A. Game-based VR

Different types of VR have different characteristics and applications. One approach for implementing VR is based on multi-player computer game engines, which provide the developers with various basic functions such as graphics rendering, sound generation, physics modeling, game logics, artificial intelligence, user interactions and networking [11,12]. The resulting VR implementations can be immersive, distributed and collaborative [13]. Some sample implementations include virtual laboratory systems [14], virtual rehabilitation systems [15, 16], training systems [17,18,19, 20] and entertainment [21, 22]. Computer games are classified into first person, third-person or a combination of these two types [23]. Moreover, they enable the users to create their own avatars and can be used as a platform for realizing interactions between virtual environments and real physical systems, thus fostering the users' sense of immersion.

B. Status of Current Integration Methods

Overview

Recently, attempts of integrating physical devices into virtual environments have begun to be reported. Among those, there are four examples for the integration of real experimental devices into virtual laboratory environments. As part of the iLab Project [24] by MIT, simulations and remote experiments from the iLab infrastructure [25] were integrated into a virtual environment that was implemented using Project Wonderland [26]. The virtual laboratory environment SecondLab [27] was developed based on Second Life [28] by the University of Deusto und subsequently used as a platform for controlling remote experiments over the WebLab-Deusto architecture [29]. 3D AutoSysLab developed by the Universidade Federal de Santa Maria based on Open Simulator [30] interfaces with real and simulated experiments [31]. A general approach for communicating between virtual environments and physical devices was developed by Stevens Institute of Technology (SIT) and a virtual laboratory environment was implemented as a pilot system with access to a remote experiment [32] using Garry's Mod (GMod) [33] and the Source game engine[34].

Below, these systems are evaluated to determine their suitability in different usage scenarios based on a series of performance criteria. The criteria include licensing, platform dependence, communication latency, scalability, reliability and user friendliness.

Licensing

It is desirable for the software implementations of game engines and the associated tools (software development kits, application programming interfaces, etc.) to be open source, free and standardized, which tends to lead to an active developer community.

All the above-mentioned game engines and associated tools are either open source and completely free or involve only a modest licensing fee. Amongst developers, Source is one of the most popular game engines, while the currently extensive base of Second Life is shrinking. In contrast, the fact that Open Wonderland and Open Simulator are developed by comparatively small communities leads to fairly slow upgrading and debugging cycles.

Platform Dependence

When linking physical devices to virtual environments, the communication techniques used should be general and compatible with other platforms for virtual environments and employ interoperable communication interfaces. The Linden Scripting Language [35] used in Second Life is not compatible with popular programming languages such as Java, C++, Visual Basic, etc. and also offers very limited support of data types and insufficient capabilities for managing complex data structures [27]. Glue code was used in SecondLab to interface the various software components, which renders this approach platform specific. In iLab and AutoSysLab, on the other hand, the approach for integrating physical devices into the virtual environment is compatible with other platforms by developing some middleware. However, the Dynamic Link Library (DLL) method presented here is not specific to one kind of game platform, which allows its adaptation to other platforms with minor modifications.

Communication Latency

The methods used for data transmission between virtual environments and physical devices should be optimized, i.e. the communication should be as fast as possible in order to achieve a better user experience. The latency in the data transmission is determined by the processors, the network and the communication protocols. However, actual data for the latency of the above-mentioned platforms have not been published yet.

Scalability

Scalability refers to the number of users that a system can service at any given time. In this respect, iLab, WebLab-Deusto and the prototype system at SIT were all developed with a scheduling function to accommodate multiple users and multiple experiments.

Reliability

Both the hardware/software components as well as the communication between the virtual environment and the physical devices should be reliable. Possible problems include software crashes, hardware component failures, communication package losses and excessive latency, among others. In order to eliminate the effects of these problems, it would be desirable to equip the system with redundancy or at least with an error alerting mechanism. There have been no reports of the assessment of the reliability of the above-mentioned VR systems yet, but some conclusions may be drawn by considering the limitations of the technology used to build these platforms. Second Life employs XML-RPC [36], which is not suitable for implementing complex services [29]. While Open Simulator is becoming more stable as it approaches release 1.0, it is still considered alpha software [30], which is lacking comprehensive system monitoring functions.

Animation Accuracy

In VR systems with integrated physical devices, the accuracy of animations of the physical behavior depends mainly on the graphics resolution and the display refresh

rate [37]. Animations developed using the Source game engine and Second Life are smooth and accurate. Also, both these systems contain sophisticated physics engines which allow for development of simulations of the physical behavior. Since SecondLab uses only Second Life as interface for manipulating physical devices, the users are merely presented with a video stream that is projected into the game environment. The animation quality in both Open Wonderland and Open Simulator is comparatively low, i.e. the resolution of the models is fairly low and motions may appear somewhat jerky [38,39].

Data Type Compatibility

It is desirable for the various game platforms to accommodate many data types, including alpha-numerical strings, formatted texts, images, audio and video streams, etc. Second Life and GMod are compatible with all the data types listed above, albeit some may require slight adaptations. Open Wonderland itself does not provide too many functions for data handling, but due to its open-source nature, various auxiliary modules created by the developer community are available [26]. The audio support in Open Simulator is currently primitive and works only partially. Similarly, Open Simulator currently does not accommodate video streaming [30].

User Friendliness

Minimal user skills should be necessary to navigate in and interact with virtual environments. Open Wonderland and Open Simulator lack support for many of the features that are common in other game platforms, which renders these systems less straightforward and self-explanatory to users who are familiar with other game platforms. Furthermore, the instant messaging in Open Simulator requires third party plug-ins and the players' names cannot be displayed, which makes it difficult for users to collaborate [30]. In SecondLab, a long specified procedure must be followed in order to achieve even simple functions.

E. Objectives of this Paper

VR systems had been the subject of active research in the mid of 1990s, but subsequently further developments in that area slowed down considerably. In addition to the limitation of current VR systems due to their purely virtual nature, this slowdown was attributable mainly to the following three reasons: VR systems were too expensive, head-mounted displays caused discomfort to the users, and the limitation to only two (or three) of the five human senses hampered the sense of realism experience by the users. In order for VR systems to reach their full potential as envisioned a couple of decades ago, these shortcomings have to be overcome.

This paper will present VR systems implemented using computer game engines, wherein the virtual environments are linked to real physical devices with a general method. This approach reduces the previously prohibitively high VR system cost significantly and at the same time replaces the problematic head-mounted displays by regular computer displays. In addition, the integration of physical devices into virtual environments has the potential for improving the users' feeling of immersion in the environments.

The remainder of this paper is divided into three sections:

- Methods for integrating virtual environments with real physical devices
- A virtual laboratory environment implemented using the Source game engine \ with real physical devices
- Potential future applications of game-based VR systems

III. INTEGRATION OF VR SYSTEMS WITH PHYSICAL DEVICES

A. Methods for Integration of VR Systems with Physical Devices

Prior Integration Platforms for VR

The concept of combining virtual environments with real physical devices is not new since augmented reality is just such kind of a VR system. Some of the first applications of this concept had been introduced for robots more than 20 years ago [40,41], and the interaction between real and virtual robots could be considered as the antecedents of modern augmented reality systems. These implementations were custom designed, device specific and used a specialized 2D display to visualize the robots. Later, 3D graphics technology was employed to develop more advanced virtual environments. Then, augmented virtual reality systems based on 3D graphics technology was widely used in medical, manufacturing and repair, annotation and visualization, robot path planning, entertainment and military aircraft applications [8]. The main characteristics of augmented reality platforms have been described as a combination of real and virtual objects, interactivity in real time and registration in 3D [8]. For instance, MS OneNote was used as a platform for integrating real (i.e. paper) and virtual (i.e. electronic) documents [42], but that system lacked a user-friendly interface. After recent advancements, augmented reality is now also used in training systems. For example, a VR system for batting practice was developed using OpenSim as the development platform [43], but this platforms limits the system to a small number of users.

VR Systems Based on Computer Game Engines

The purpose of computer game engines is to provide a suite of development tools for games that make the common components of the computer games reusable and adaptable [11,12]. When computer game engines are employed to implement VR systems that are linked to real physical devices, the exploitation of the middleware that is part of the game engine allows for efficient and fast development (see for instance [44]). Conventionally, the submodules of a game engine responsible for rendering, physics simulation, environment modeling, texturing and networking are denoted as middleware. Alternatively, the entire game engine could be considered a middleware [45]. In order to link game-based virtual environments with real physical devices, the game engine's software development kit (SDK) and application programming interface (API) must be compatible with other software applications and hardware platforms, such as CAD/CAM packages, LabVIEW, MATLAB, hardware peripherals or other physical devices (see Figure 1).

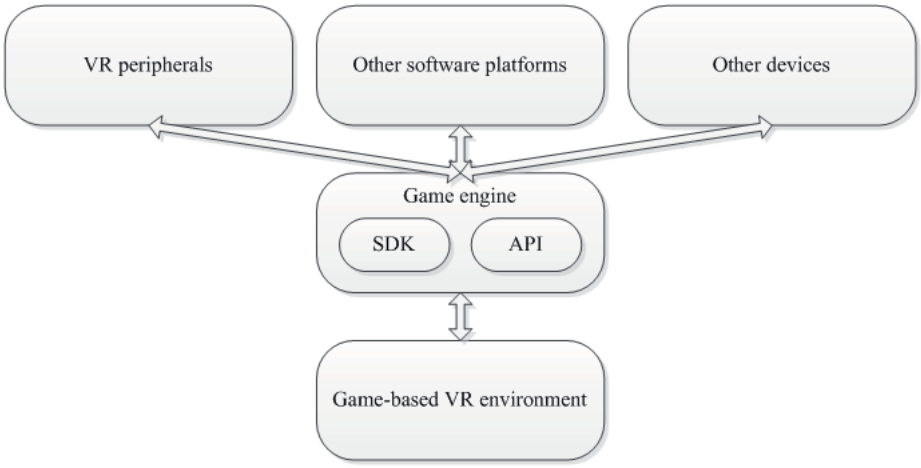

Figure 1. Integration of game engine with other applications

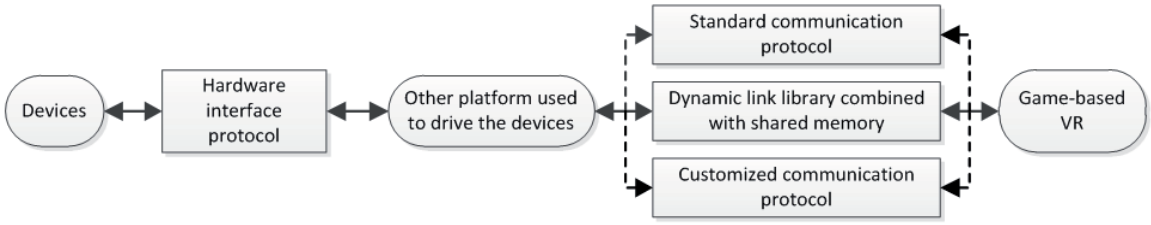

Figure 2. Methods of communication between game-based VR systems and physical devices

C. *Integration Using Dynamic Link Libraries*

When linking virtual environments implemented using computer game engines, the communication between the game engine and the other platforms for controlling the real physical devices can be realized either by standardized or platform-specific communication protocols or alternatively by mixed-language programming combined with shared memory. The use of standardized communication protocols is preferred when they are supported by all the different platforms to be linked [46]. In some instances, the platforms to be linked may not all support a common standardized communication protocol [47], which then necessitates the development of a customized, platform-specific communication protocol. The latter approach is costly, and therefore, it is preferable to resort to mixed-language programming, where source codes written in different programming languages are compiled together and can be called back by different platforms. In the present work, Dynamic Link Libraries (DLLs) under MS Windows were employed because most of the present computer game engines support the concept of DLLs (Dynamically Linked 'Shared Object' Libraries under Linux), and so do most of the platforms one may be interested in linking to game engines, for instance MATLAB, LabVIEW, Java, Basic, C/C++, etc. Figure 2 illustrates the three different methods. In addition, the DLL-based approach allows for faster communication between a game engine and a physical device than standard communication protocols such as TCP [48].

In principle, a DLL file simply represents a shared library in the MS Windows or Linux operating systems. It may contain data, functions, code, binary files for the operating system, custom resources or any combination thereof. Furthermore, its different contents have different extensions (for MS Windows *.dll, *.exe, *.drv, *.fon, etc.; for Linux *.so). The subsequent discussions in this paper will be limited to MS Windows. For the application presented here, *.dll files are sufficient for implementing the communication between the game-based VR system and the other platforms. This communication involves only data, functions and codes, which can all be packed into a *.dll file.

B. *Sample Implementation for Integration of Platforms*

Below, a sample implementation based on mixed-language programming is described, wherein *.dll files are used to link a specific game engine (GMod) endowed with a scripting language (Lua) with a specific platform (LabVIEW). Subsequently, a game-based laboratory environment is presented as an application of this approach.

In this method, GMod and LabVIEW are two different processes running under the MS Windows operating system. Then, the same *.dll files are shared between these different processes, i.e. they can be called simultaneously by these processes and they are used as carriers for sharing data, functions and codes.

The motivation for using this technique is that the *.dll files compiled by C++ under the MS Windows operating system can be loaded by both LabVIEW and the GMod game server. In LabVIEW 6.x (or later versions), *.dll files can be called by the call library function node (CLFN). Note that the CLFN used to be referred to as code interface node (CIN) before, but LabVIEW no longer supports this node [49]. In GMod, the extended interface (i.e. the API of GMod), which is used to load modules in the form of *.dll files, is denoted as GModInterface.

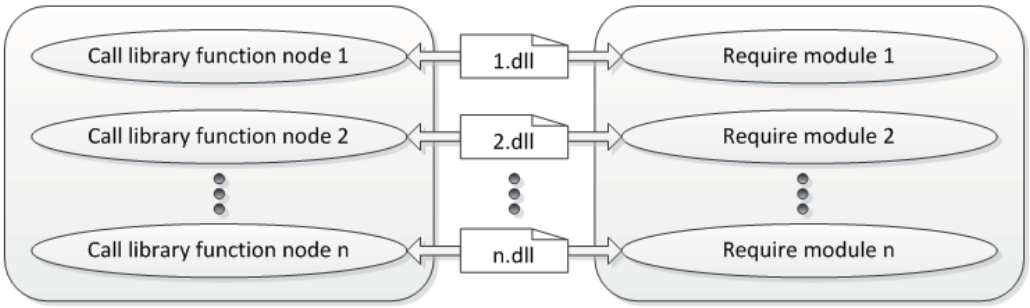

Figure 3. Functions supporting communication between LabVIEW and GMod

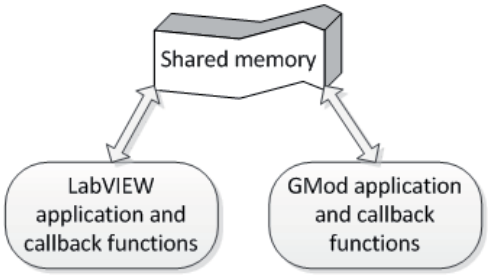

Figure 4. Architecture of *.dll file

This structure enables Lua scripts to communicate with the *.dll files. Then, the GMod interface uses Lua's C application programming interface (C API) to embed data or functions into the *.dll files. Furthermore, specific functions used in LabVIEW can also be compiled in the same shared *.dll file. After this step, any specific *.dll file can be called by both GMod and LabVIEW in different processes [50]. A simple flow chart of the communications between the GMod and LabVIEW using *.dll files is depicted in Figure 3.

The above structure can only realize the simple callback of data and functions that are packaged in *.dll files. However, the problem of interest here is to interchange data between different platforms and different processes. In order to solve this problem, a shared memory that can give simultaneous access to LabVIEW and GMod (which represent different processes) is allocated. This shared memory avoids unnecessary copies of the instances of libraries which must be used as static libraries among the processes, thus improving the efficiency in passing data between different platforms [51]. The architecture of *.dll files that are shared between LabVIEW and GMod is illustrated in Figure 4.

In order to illustrate this method, a collaborative virtual laboratory environment developed previously using the multi-player computer game engine 'Source' and GMod [52] was linked to real physical laboratory setups as described in the next section.

IV. GAME-BASED COLLABORATIVE VIRTUAL LABORATORY ENVIRONMENT WITH INTEGRATED PHYSICAL DEVICES

A. General Remarks

The collaborative laboratory environment presented here was implemented by extending the game environment GMod, which is based on the multi-player game engine 'Source'. This extension included the creation of models, functions and a map as well as the establishment of a link to actual physical devices for conducting educational remote experiments as discussed in detail in the subsequent sections. Figure 5 describes the development process of this system and explains the relationship between the concept of VR and the game-based virtual laboratory.

B. Remote and Virtual Laboratories

The term remote laboratory refers to the use of telecommunications to conduct real experiments from a remote location. In recent years, remote laboratories have become an alternative and/or complement to traditional hands-on laboratories associated with science and engineering curricula at many higher education institutions worldwide. Significant contributions to this area include iLabs [24,53], Labshare [54], LiLa [55], Netlab [56], WebLab [57], iLough-Lab [58] and VISIR [59]. Other noteworthy remote laboratories are described elsewhere [60,61,62,63,64]. While remote laboratories have been found to have several advantages over conventional hands-on laboratories (e.g. increased accessibility, improved safety and security, reduced operating costs), they are most suitable for illustrating conceptual knowledge as opposed to instilling design skills, and they fail to provide the students with experiences in setting up, trouble shooting and debugging experimental equipment [65,66]. Virtual or simulated laboratories are software-based imitations of real experiments. In various studies, they have been found to be good substitutes for hands-on experiments in teaching abstract concepts and their applications [67]. Virtual laboratories also provide advantages such as low operating costs, reduced time for set-up and tear-down of the experiments, and they create an active mode of learning [65]. However, they are believed to have disadvantages due to the disconnection between the real and virtual worlds [65] and often do not include the imperfections encountered in real systems.

C. Concept of Game-based Virtual Environments

Game-based virtual environments are designed to closely simulate aspects of real or fictional scenarios and recently have begun to be used for educational and training purposes, for instance in medical training [68], military training [69], sports training [43], management simulations [70], and life simulations [28], etc. Research on game-based learning has demonstrated that games increase the users' intrinsic motivation through fantasy, control, challenge, curiosity and competition [71].

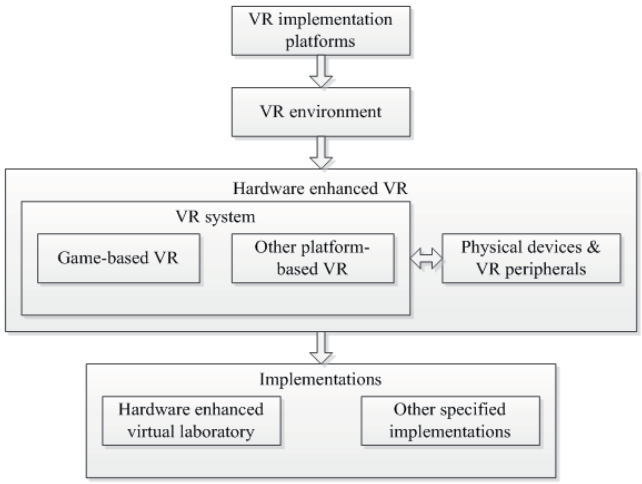

Figure 5. Schematic of system development process

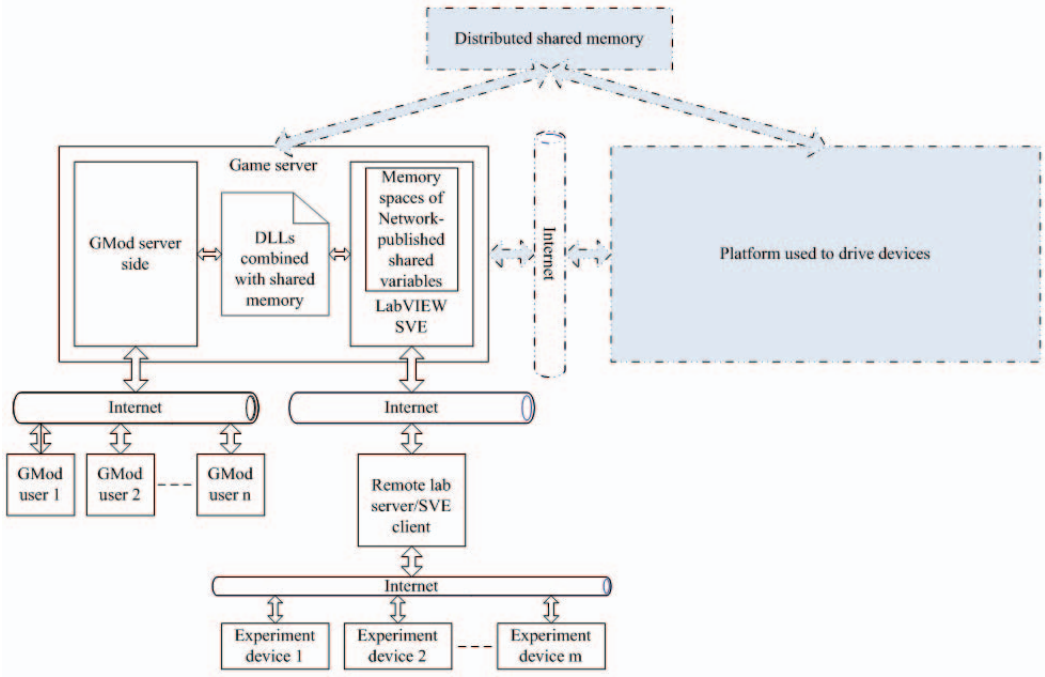

Figure 6. System architecture

Also, by providing visual and auditory feedback, games increase the interactivity between users and environments and enable the users to learn by playing [72]. Without doubt, playing is a powerful and pervasive method of learning, by which ideas are tested, new skills are developed and people participate in new social roles [71].

D. System Architecture of Collaborative Virtual Laboratory Environment

The system architecture of the collaborative laboratory environment presented here is shown in Figure 6. In this system, the communication between different platforms is realized by DLLs combined with shared memory or by distributed shared memory (DSM). If the different processes of the platforms are running in different computers located throughout the network, then DSM must be employed as indicated by the parts bounded by dashed-double-dotted lines. On the other hand, if all of the processes are executed in one computer, then DLLs combined with shared memory should be used rather than DSM because the latter requires communication between different processes through the network, which is inherently slow [73]. In the application presented here, all processes (i.e. GMod and LabVIEW) are executed in one computer (i.e. game server), and their integration is discussed in detail below. The users access the client side of GMod. Then, the information is passed on to the server side of GMod by the network component of the game engine. Finally, the server side of GMod exchanges data with the server side of LabVIEW's shared-variable engine (SVE) through DLLs combined with shared memory. Then, the network-published shared variables pass data back and forth between the game sever and the remote laboratory server/SVE client.

```
require (".DLL");
set of variables;
function_1 (argument_1, …, argument _i_1);
function_2 (argument_2, …, argument _i_2);
…
Function_n (argument_ 1, …, argument_i_n);
return values_1 = function_n+1 (argument_1, …, argument_ i_n+1);
return values_2 = function_n+2 (argument_ 1, …, argument_ i_n+2);
…
return values_m = functio_n_n+m (argument_1, …, argument_i_n+m);
```

Figure 7. Lua script for passing data between GMod and shared memory

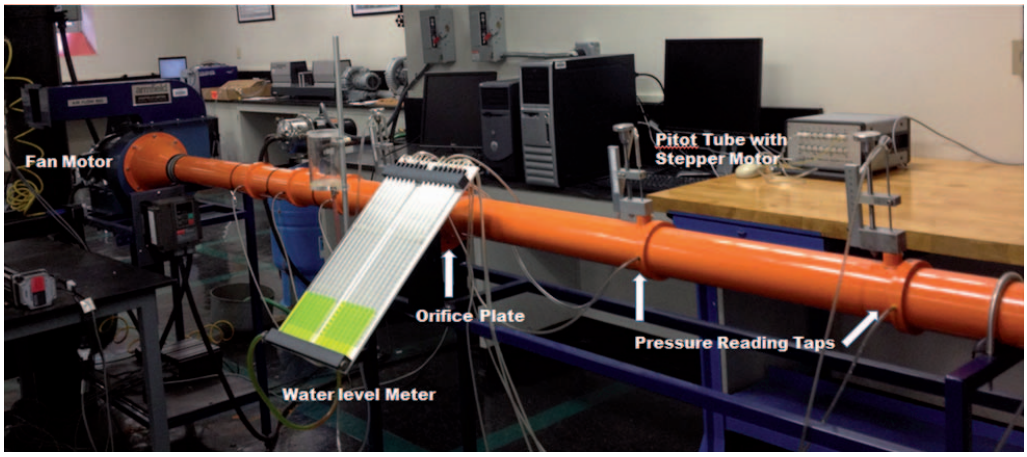

Figure 8. Air flow rig (courtesy of Armfield Inc.)

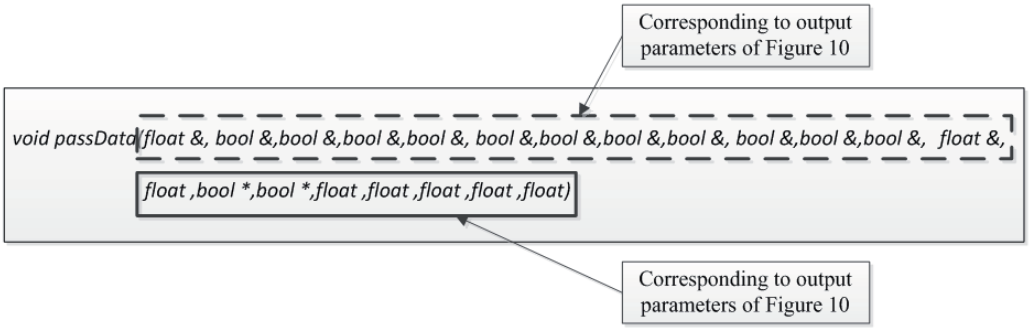

Figure 9. Prototype of passData() function for data exchange between GMod and LabVIEW's SVE

The remote laboratory server/SVE client is used to connect with the experimental devices through the Internet with another set of network-published shared variables. Although all of the devices can be connected directly with the LabVIEW SVE, they are managed by the remote laboratory server. This two-level network architecture facilitates the management of all data from the devices.

The communication between LabVIEW and GMod is summarized in Figure 3. A *.lua script file is loaded automatically when starting GMod. This file is composed of a basic set of variables and the necessary functions to read/write data from shared memory. Among these functions, some that do not have return values are used to pass data from GMod to the shared memory. The other functions with return values are used to get data from shared memory. The pseudo code used to explain how to call a *.dll file and how to use these variables and functions in a *.lua script file is listed in Figure 7.

Functions 1 to n+m are registered functions that reside in the *.dll file. Moreover, according to the conventions of the GMod interface, if a *.dll file is taken as one of the modules of the GMod game server, the name of this *.dll file must be in the format gm_*.dll. Then, GMod can load this module using the 'require()' function provided by the Lua scripting language.

E. Sample Experiment

Experimental Setup

A multi-user game-based virtual laboratory environment with an integrated remote flow-development apparatus was developed as a pilot application at SIT. This implementation uses a commercially available air flow rig by Armfield Inc. (see Figure 8). Experiments were designed for measuring the important characteristics of industrial air distribution systems as well as to show how certain basic principles of fluid mechanics are applied to analyze flows in ducts and jets.

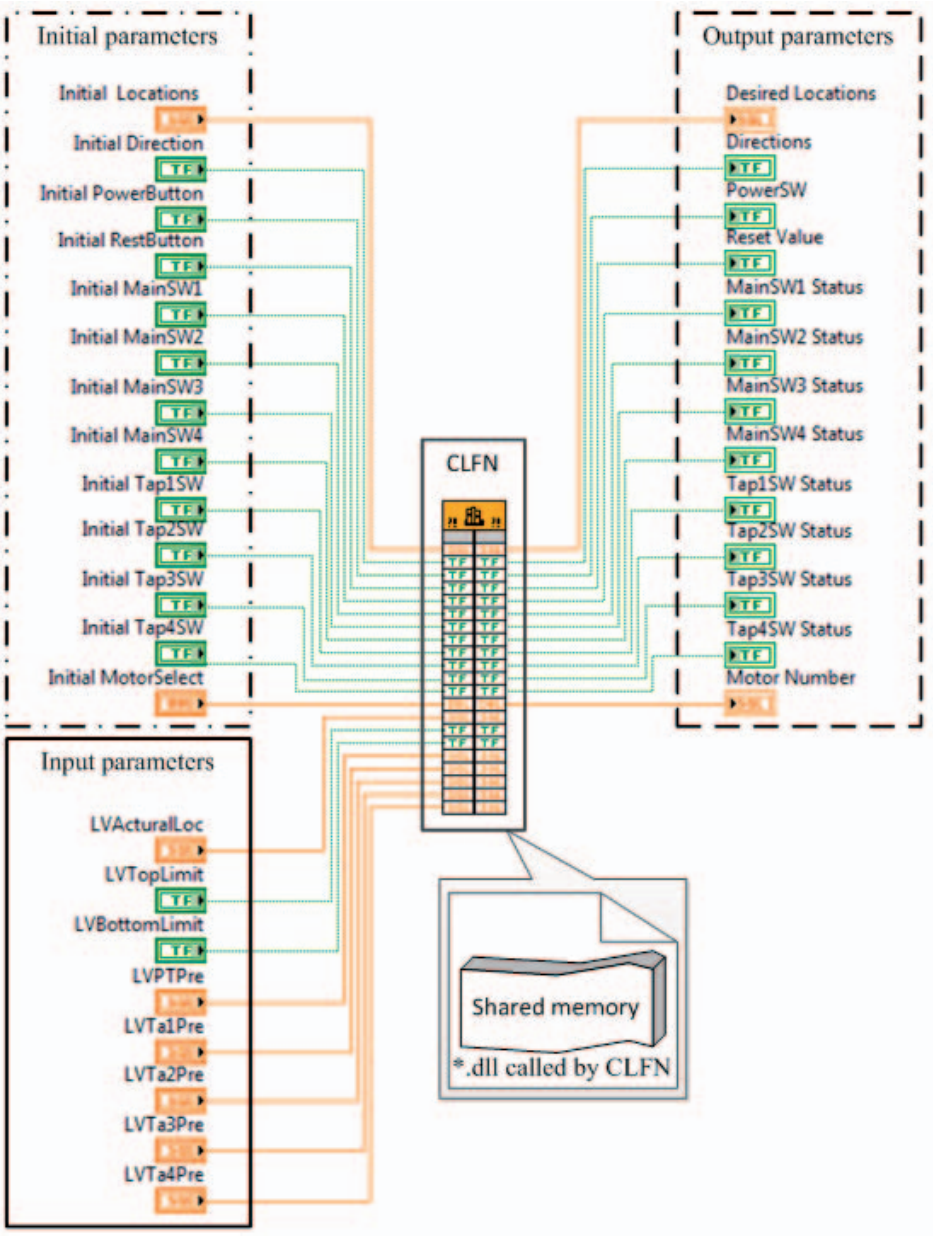

Figure 10. Calling DLL in LabVIEW with CLFN

In this experimental setup, motor-controlled Pitot tubes are employed to measure the pressure distributions at various cross sections and at the outlet of the flow pipe. One of the experiments enables the calibration of a flow meter based on an orifice plate that is inserted into the air stream. An orifice plate is a precisely measurable obstruction that narrows the pipe, thus constricting the flowing substance. The experiment also allows the exploration of the flow development in a straight flow pipe and the determination of the free-flow velocity profile at the pipe's outlet [74].

Middleware

In the demonstrated implementation, only one *.dll file is compiled for use with the laboratory environment since the data and the functions to be passed are not very complicated. However, more than one *.dll file can be compiled and uploaded at a time as depicted in Figure 3. In order to explain how the system works, Figure 10 shows a piece of the LabVIEW graphics program that is used by the CLFN to call the *.dll file in which the corresponding prototype of the function is imbedded. This function is used to exchange data between GMod and LabVIEW's SVE, and its prototype is shown in Figure 9.

As shown in Figure 10, the initial parameters (see Table 1) are used to set the initial values of the output parameters. The input parameters (see

Table 2) are used to transmit the experimental data from the real laboratory to GMod and to then configure the animation of the corresponding models of the experimental devices in GMod. These input parameters correspond to the last eight arguments in the passData() function shown in Figure 9. The output parameters (see

Table 3) are used to pass data from GMod to LabVIEW in order to control the experimental devices. These output parameters correspond to the first thirteen arguments in the passData() function shown in Figure 9.

TABLE 1.

INITIAL PARAMETERS APPEARING IN FIGURE 10

Name	Description	Name	Description
Initial Locations	Jet piston initial position	Initial MainSW4	Main switch 4 initial status
Initial Direction	Motor initial rotation direction	Initial Tap1SW	Tap 1 switch initial status
Initial PowerButton	Power switch initial status	Initial Tap2SW	Tap 2 switch initial status
Initial ResetButton	Reset switch initial status	Initial Tap3SW	Tap 3 switch initial status
Initial MainSW1	Main switch 1 initial status	Initial Tap4SW	Tap 4 switch initial status
Initial MainSW2	Main switch 2 initial status	Initial MotorSelect	Initial selected motor
Initial MainSW3	Main switch 3 initial status		

TABLE 2.

INPUT PARAMETERS APPEARING IN FIGURE 10

Name	Description	Name	Description
LVActualLoc	Jet piston actual position	LVTa1Pre	Tap 1 pressure
LVTopLimit	Jet piston top limit position	LVTa2Pre	Tap 2 pressure
LVBottomLimit	Jet piston top bottom position	LVTa3Pre	Tap 3 pressure
LVPTPre	Pit tub pressure	LVTa4Pre	Tap 4 pressure

TABLE 3.
OUTPUT PARAMETERS APPEARING IN FIGURE 10

Name	Description	Name	Description
Desired Locations	Control jet piston position	MainSW4 Status	Main switch 4 status
Directions	Motor rotation direction	Tap1SW Status	Tap 1 switch status
PowerSW	Power switch	Tap2SW Status	Tap 2 switch status
Reset Value	System reset	Tap3SW Status	Tap 3 switch status
MainSW1 Status	Main switch 1 status	Tap4SW Status	Tap 4 switch status
MainSW2 Status	Main switch 2 status	Motor Number	Motor selection
MainSW3 Status	Main switch 3 status		

Development of Laboratory Environment

The virtual laboratory environment presented here was created using the 'Source' game engine, which provides all functions needed to develop a realistic virtual environment. In addition, it provides extensive support via its 'Source' SDK, which includes the 'Hammer' map editor. The latter enables users to create and edit the virtual map of the game environment [10].

The models of the experimental components of the virtual laboratory environment were created based on custom 3D CAD models of the real physical equipment. Such custom models can be built using third-party 3D modeling software such as 3ds Max [75] or SolidWorks [76]. The flow rig assembly models of the experiment described here were created in SolidWorks and then converted into a file format that is compatible with GMod.

The 'Source' game engine employs the 'Havok' physics engine [77] to model real-world phenomena based on Newtonian physics in order to achieve realistic interactions between objects in the virtual environments [14]. Furthermore, certain features of 'Havok' were also used to implement some functionality of the virtual laboratory environment, such as collision detection during the assembly of components into experimental systems and the support of realistic animations of the experimental devices. GMod provides an extensive Lua library, which enables programmers to create game rules and game entities and to define the behavior of certain game characters and objects [78]. In the virtual laboratory environment, the Lua scripting language adopted by GMod was also used to create tools and game modifications.

Experimental Procedure

In real hands-on experiments using the flow rig, the experimental apparatus has already been set up when the students arrive in the laboratory. Thus, the students just need to start up the fan motor, adjust its speed, select the Pitot tube to be used for the required pressure measurements, adjust its position using a step motor, read off the values of the water-level meter and calculate the corresponding pressure values. Then, this procedure can be repeated several times for different Pitot tube locations and/or positions as well as for different fan speeds.

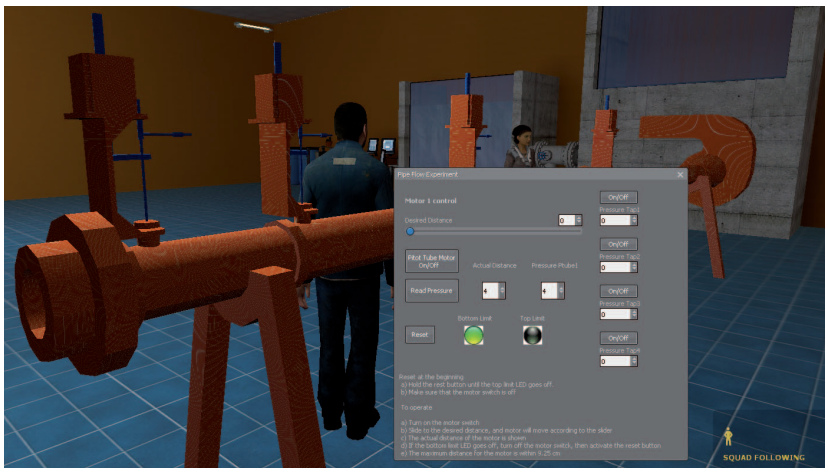

Figure 11. 3D virtual model of flow rig apparatus

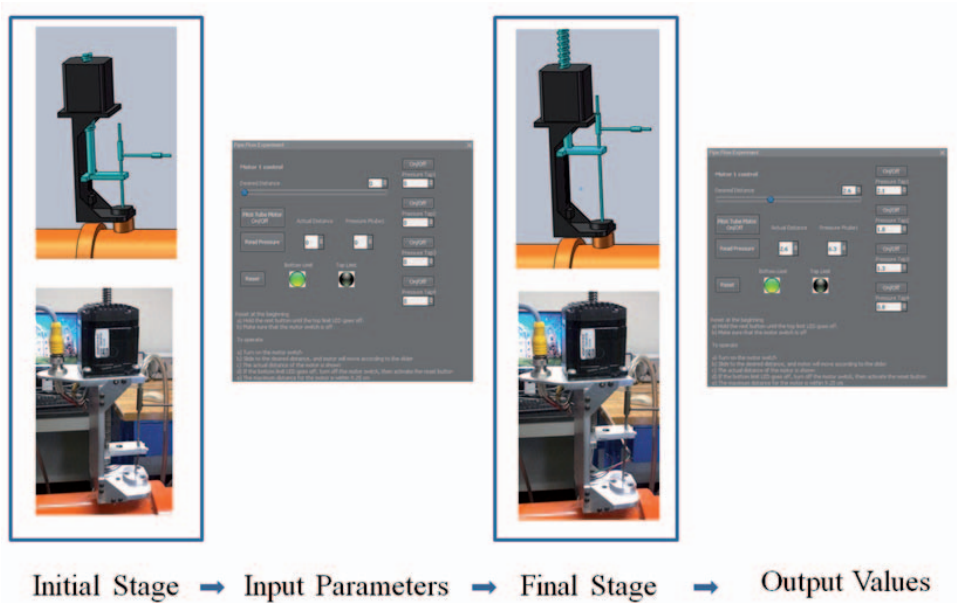

Figure 12. Simulations linked to a real device

However, in the virtual laboratory environment described here, the students, the instructor and the teaching assistant are represented as avatars. They interact with a virtual flow rig apparatus that represents the real physical device. The positions of the Pitot tubes can be observed based on a real-time animation within the virtual laboratory environment. This animation is driven by data acquired from the physical devices or based on the video stream from a webcam that is located at the real flow rig apparatus (see Figure 12).

To achieve the intended learning outcomes, students should be able to perform the following laboratory exercises:

- Assemble the components of the flow rig setup into a functioning system (including a base, a flow pipe with built-in fan motor, fan and diffuser, step motors, Pitot tubes, pressure reading taps and an orifice plate if needed).
- Input the initial parameters (including fan speed, Pitot tube selection and position)
- Press the RUN button to start the acquisition of the pressure data (see Figure 11)
- Record the pressure data (either electronically into a file or manually on paper)
- If needed, repeat Steps 2 to 4 with different input parameters in order to obtain entire pressure distributions at the locations of interest
- If needed, modify the experimental setup by inserting an orifice plate into the flow and repeat Steps 1 to 5.

V. OTHER POTENTIAL APPLICATIONS

A variety of VR applications can be found in the literature (see for instance [79 ,8]). Some of these applications that could be implemented using the platform presented above are discussed below. Because of the use of the mixed programming technique for integrating the physical devices and the VR system, the platform described above (possibly with some minor modifications) can be made compatible with most hardware interface drivers, i.e. it is capable of supporting most peripheral

devices used to connect the real and virtual worlds. In addition, the multiplayer game-based platform lets all users engage in group collaboration and relieves the users' tension resulting from some serious game applications such as virtual rehabilitation, military training, etc.

Virtual rehabilitation makes it possible to implement therapeutic interventions locally or at a distance with the help of VR systems. VR therapy systems have been applied to various patient populations, including musculo-skeletal, post-stroke and cognitively impaired patients [15]. The benefits and challenges of virtual rehabilitation were discussed in more detail in [15]. The concept of VR rehabilitation for post-stroke patients was introduced earlier [80,81,82], and more recently, the SeeMe system [83] was presented. The procedures of rehabilitation are tedious and painful, and therefore improving the patients' motivation and keeping the patients interested in the therapy is considered critical in order to achieve good clinical outcomes. Although products such as SeeMe were developed based on first-person game environments, it is unavoidable for patients to feel lonely in such single-person games. Multiplayer game-based VR platforms have the potential for solving this problem, resulting from the inherent entertainment characteristics of games. In VR therapy systems, doctors, therapists and patients can interact in the same scenario: the patients can get instructions from the doctors and therapists, the therapists can consult with the doctors, and the therapists and doctors can get feedback from the patients. In fact, all of these activities can happen in real time. At the same time, certain patient groups could also use rehabilitation devices locally while the therapists operate these devices remotely through this platform.

Similarly, various virtual athletic training systems [84,85] and virtual military training system [86,87] have been developed using VR. The cost of these systems is often significant since the implementation of a user-friendly virtual environment is a complex proposition unless it is based on a ready-made platform. Fortunately, VR based on game engines can shorten the development time and cost significantly. In addition, multiplayer game-based VR platforms also provide an effective environment for group training (e.g. team sports).

Finally, VR is also often used as a tool for treating psychiatric and psychological diseases, for example various phobias and disorders [88,89,90,91,92]. Although VR applications in the realm of psychiatry and psychology remain controversial, some results are promising [93]. For instance, a virtual sandbox treatment was introduced [94], and it was argued that play therapy including a virtual sand box is helpful for children with social phobias [95]. A shortcoming is that in a virtual sand box based on both augmented reality and immersive VR, the users play only with virtual characters. In systems based on the platform presented here, other persons could join in the same environment by creating their own avatars and interact with the users who need help. By immersing the users with problems in environments that feel more real, the effectiveness of the treatment could be improved significantly.

VI. CONCLUSIONS AND FUTURE RESEARCH

In this paper, the current state of VR and the integration of the real and virtual worlds were studied and the advantages and disadvantages of current technologies in game-based VR were compared. In addition, a general method for integrating VR and physical activities was proposed. This method is based on the mixed programming technique and employs callable DLLs for the communication between the VR and other application platforms. In order to demonstrate the efficiency of this method, a multiplayer game-based virtual laboratory environment was implemented as one of the instances of VR. This system was integrated with a prototype of a remotely controllable, real-time air flow laboratory apparatus. This system has proven to be a promising, economical and efficient way to overcome the difficulties in integrating physical devices into VR systems.

In future work, some other challenges associated with the development of VR should be addressed, including for instance the creation of virtual environments in real time, the seamless integration of different platforms and systems, the facilitation of more efficient interactions between humans and virtual environments, and the integration of 3D sound synthesis, stereoscopic 3D display technologies and perception feedback. The integration of VR and physical activities has tremendous potential in diverse fields such as medical, athletic and military training and virtual rehabilitation, but future generations of these kinds of applications could benefit significantly from the integration of real-time video streams into virtual environments, the real-time generation of avatars by the users themselves and further improvements in the immersive perception of the users.

ACKNOWLEDGMENTS

This multi-disciplinary research project was carried out at Stevens Institute of Technology with funding from NSF CCLI Grant No. 0817463. This support is gratefully acknowledged. Also, the authors wish to thank Dr. El-Sayed Aziz for many stimulating discussions on the topic.

REFERENCES

[1] McLellan, H., 2001, "Virtual Realities", McLellan Wyatt Digital.

[2] Gaggioli, A. & Breinin, R., 2001, "Communications through Virtual Technology, Identity Community and Technology in the Internet Age, Edited by G. Riva and F. Davide, IOS Press, Amsterdam.

[3] Dilwort, J., 2010, "Realistic virtual reality and perception", *Philosophical Psychology*, Vol. 23, pp. 23-42. http://dx.doi.org/10.1080/09515080903533942

[4] Brady, S. & O'Sullivan, C., 1999, "3D training environments – VRML and its use in interactive task-based simulations, *Technical Report*, TCD-CS-1999-20, March 1999.

[5] Zheng, J., 2000, "VR interfaces for conceptual design using geometric modeling techniques", *Ph.D. Disse*rtation, The University of Hong Kong, May 2000.

[6] Ausburn, L. J. & Ausburn, F. B., 2004, "Desktop virtual reality: a powerful new technology for teaching and research in industrial teacher education", *Journal of Industrial Teacher Education*, Vol. 41, No. 4, pp. 1-16.

[7] Freund, E. & Roßmann, J., 2003, "Distributed virtual reality: system concepts for cooperative training and commanding in virtual worlds", *Journal of Systemics, Cybernetics and Informatics*, Vol. 1, No. 1, pp. 1690-4524.

[8] Azuma, R. T., 1997, "A survey of augmented reality", *Presence: Teleoperators and Virtual Environments*, Vol. 6, No. 4, pp. 355-386.

[9] Silva, R., de Oliveira, J. C. & Giraldi, G. A., "Introduction to augmented reality", *LNCC Research Report*, No. 25/2003, National Laboratory for Scientific Computation, ISSN: 0101 6113.

[10] Chang, Y., Aziz, E.-S., Esche, S. K. & Chassapis, C., 2012, "A game-based laboratory for gear design", *Computers in Education Journal*, Vol. 22, No. 1.

[11] Baba, S. A., Hussain, H. & Embi, Z. C., 2007, "An overview of parameters of game engine", IEEE Multidisciplinary Engineering Education Magazine, Vol. 2, No. 3, pp. 10-12.

[12] Thorn, A., 2010, "Game Engine Design and Implementation", Chapter 1, 1st Ed., Jones & Bartlett Publishers.

[13] Aziz, E.-S., Chang, Y., Tumkor, S., Esche, S. K. & Chassapis, C., 2010, "Adapting computer game technology to support engineering laboratories", *Proceedings of the ASME International Mechanical Engineering Conference & Exposition*, November 12-18, Vancouver, British Columbia, Canada.

[14] Chang, Y., Aziz, E.-S., Esche, S. K. & Chassapis, C., 2011, "Overcoming the limitations of current online laboratory systems using game-based virtual environments", *Proceedings of the ASME 2011 International Mechanical Engineering Congress & Exposition*, Denver, Colorado, USA.

[15] Burdea, G. C., 2002, "Key note address: virtual rehabilitation – benefits and challenges", *Proceedings of the First International Workshop on Virtual Reality Rehabilitation (Mental Health, Neurological, Physical, Vocational)*, Lausanne, Switzerland, November 7-8, 2002, pp. 1-11.

[16] Jack, D., Boian, R., Merians, A. S., Tremaine, M., Burdea, G. C., Adamovich, S. V., Recce, M. & Poizner, H., 2001, "Virtual reality-enhanced stroke rehabilitation", IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 9, No. 3, pp. 308-318. http://dx.doi.org/10.1109/7333.948460

[17] Harrison, G. W., Haruvy, E. & Rutström. E. E., 2011, "Remarks on virtual world and virtual reality experiments", *Southern Economic Journal*, Vol. 78, No. 1, pp. 87-94. http://dx.doi.org/10.4284/0038-4038-78.1.87

[18] Fox, J., Arena, D. & Bailenson, J. N., 2009, "Virtual reality – a survival guide for the social scientist", *Journal of Media Psychology*, Vol. 21, No. 3, pp. 95-113. http://dx.doi.org/10.1027/1864-1105.21.3.95

[19] "The Virtual Football Trainer", http://www-vrl.umich.edu/project/football/index.html, accessed in August 2012.

[20] Stepan, V. & Zara, J., 2002, "Teaching tennis in virtual environment", Proceedings of the Spring Conference on Computer Graphics, Budmerice, Slovakia, pp. 49-54.

[21] Olanda, R., Pérez, M., Morillo, P., Fernández, M. & Casas, S., 2006, "Entertainment virtual reality system for simulation of spaceflights over the surface of the planet Mars", *Proceeding of VRST '06 Proceedings of the ACM symposium on Virtual reality software and technology, Limassol, Cyprus*, pp. 123-132.

[22] Craig, A. B., Sherman W. R. & Will J. D., 2009, "Developing virtual reality applications: foundations of effective design", Chapter 9, 1st Ed., (August 7, 2009), Morgan Kaufmann Publishers.

[23] Jacobson, J. & Lewis, M., 2005, "Game engine virtual reality with CaveUT", *Computer*, Vol. 38, No. 4, pp. 79-82. http://dx.doi.org/10.1109/MC.2005.126

[24] Harward, V. J., del Alamo, J. A., Choudhary, V. S., deLong, K., Hardison, J. L., Lerman, S. R., Northridge, J., Talavera, D., Varadharajan, C., Wang, S., Tehia, K. & Zych, D., 2004, "iLab: a scalable architecture for sharing online experiments," *Proceedings of the International Conference on Engineering Education*, Gainesville, Florida, USA, October 16-21, 2004.

[25] Scheucher, T., Bailey, P. H., Gütl, C. & Harward, V. J., 2009, "Collaborative virtual 3D environment for Internet-accessible physics experiments", *International Journal of Online Engineering*, Vol. 5, No. 1, pp. 65-71.

[26] Sun Microsystems, Inc., Project Wonderland: Toolkit for Building 3D Virtual Worlds. http://openwonderland.org/, accessed in August 2012.

[27] García-Zubia, J., Irurzun, J., Angulo, I., Hernández, U., Castro, M., Sancristobal, E., Orduña, P. & Ruiz-de-Garibay, J., 2010. "SecondLab: a remote laboratory under Second Life", *Proceedings of the IEEE EDUCON 2010 Conference*, Madrid, Spain, April 14-16, 2010, pp. 351-356.

[28] Second Life, from Linden Research, Inc., http://secondlife.com/, accessed in August 2012.

[29] García-Zubia, J., Irurzun, J., Angulo, I., Orduña, P., Ruiz-de-Garibay, J., Hernández, U. & Castro, M., 2010, "Developing a Second-Life-based remote lab over the WebLab-Deusto architecture", *Proceedings of the Remote Engineering and Virtual Instrumentation Conference* (REV2010), Stockholm, Sweden, June 29 - July 2, 2010, pp. 171-176. http://dx.doi.org/10.1109/EDUCON.2010.5492556

[30] OpenSimulator, http://opensimulator.org/wiki/Main_Page, accessed in August 2012.

[31] Schaf, F. M., Paladini, S. & Pereira, C. E., 2012, "3D AutoSysLab prototype - a social, immersive and mixed reality approach for collaborative learning environments", *Proceedings of the 2012 IEEE Global Engineering Education Conference*, pp. 1-9.

[32] Tumkor, S., Zhang, Z., Zhang, M., Chang, Y., Esche, S. K. & Chassapis, C., (2012), "Integration of a real-time remote experiment into a multi-player game laboratory environment", Paper accepted for presentation at the *ASME International Mechanical Engineering Conference & Exposition IMECE'12*, Houston, Texas, USA, November 9-15, 2012.

[33] Garry's Mod, developed by Garry Newman, http://garrysmod.com/, accessed in August 2012.

[34] Source Engine, developed by Valve Corporation, http://source.valvesoftware.com/, accessed in August 2012.

[35] Description of Linden Scripting Language from Wikipedia, http://en.wikipedia.org/wiki/Linden_Scripting_Language, accessed in August 2012.

[36] XML-RPC, from Wikipedia, http://en.wikipedia.org/wiki/XML-RPC, accessed in August 2012.

[37] Rekapalli, P. V., Martınez, J. C. & Kamat, V. R., 2009, "Algorithms for accurate three-dimensional scene graph updates in high speed animations of simulated construction operations", *Computer-Aided Civil and Infrastructure Engineering*, Vol. 24, No. 3, pp. 186-198. http://dx.doi.org/10.1111/j.1467-8667.2008.00565.x

[38] Demonstration of Open Wonderland, http://vimeo.com/6581845, accessed in August 2012.

[39] Open Simulator screenshots: http://opensimulator.org/wiki/Screenshots, accessed in August 2012.

[40] Bejczy, A. K., Kim, W. S. & Venema, S. C., 1990, "The phantom robot: predictive displays for teleoperation with time delay", *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, Cincinnati, Ohio, USA, May 13-18, 1990, Vol. 1, pp. 546-551. http://dx.doi.org/10.1109/ROBOT.1990.126037

[41] Milgram, P., Zhai, S., Drascic, D. & Grodski, J. J., 1993, "Applications of augmented reality for human-robot communication", *1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Yokohama, Japan, July 26-30, 1993, Vol. 3, pp. 1467-1472.

[42] Seifried, T., Jervis, M., Haller, M., Masoodian, M. & Villa, N., 2008, "Integration of virtual and real document organization", *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction*, Bonn, Germany, February 18-20, 2008, pp. 81-88. http://dx.doi.org/10.1145/1347390.1347410

[43] Syamsuddin, M. R. & Kwon, Y. M., 2010, "Research on virtual world and real world integration for batting practice", *Proceedings of the 2010 International Symposium on Ubiquitous Virtual Reality (ISUVR)*, Gwangju, South Korea, July 7-10, 2010, pp. 44-47. http://dx.doi.org/10.1109/ISUVR.2010.21

[44] van Oijen, J., Vanhée, L. & Dignum, F., 2011, "CIGA: A middleware for intelligent agents in virtual environments", Proceedings of the 3rd International Workshop on Agents for Education, Games and Simulations AAMAS'11, Taipei, Taiwan, May 2-6, 2011.

[45] Yip, M. K., Liu, E. S., Cheung, M. K., Lung, R. M. & Yu, G., "Design decisions in game middleware development: experiences from lucid platform", *Proceedings of the 1st International Symposium on Game's Science, Art, Education and Applications*, Taipei, Taiwan, December 2005.

[46] Microsoft Patterns & Practices Team, 2009, "Microsoft® application architecture guide", 2nd Ed., Microsoft Press, Chapter 18.

[47] Brutzman, D., Zyda, M., Watsen, K. & Macedonia, M., 1997, "Virtual reality transfer protocol (VRTP) design rationale", *Proceedings of the Sixth IEEE Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Cambridge, Massachusetts, USA, June 18-20, 1997, pp. 179-186.

[48] Mitchell M., Oldham J. & Samuel A., 2001, "Advanced Linux programming", Chapter 5, 1st Ed., Sams Publisher.

[49] LabVIEW 2010 help, available on the Web , http://zone.ni.com/reference/en-XX/help/371361G-01/lvexcodeconcepts/cins_vs_clf_nodes/, accessed in August 2012.

[50] Ierusalimschy, R., 2006, *"Programming in LUA"*, 2nd Ed., Lua.org.

[51] Windows Developer Center, July 2012, available on the Web, http://msdn.microsoft.com/en-us/library/windows/desktop/ee663297(v= vs.85).aspx, accessed in August 2012.

[52] Aziz, E.-S., Esche, S. K. & Chassapis, C., 2010, "An interactive game-based engineering laboratory', *World Transactions on Engineering and Technology Education*, Vol. 8, No. 2, pp. 131-136.

[53] iLab Group at the University of Queensland's Center for Educational Innovation & Technology, http://ceit.uq.edu.au/content/ilabs-group, accessed in August 2012.

[54] Lowe, D., Berry, C., Murray, S. & Lindsay, E., 2009, "Adapting a remote laboratory architecture to support collaboration and supervision", *Proceedings of 6th International Conference on Remote Engineering and Virtual Instrumentation*, Bridgeport, Connecticut, USA, June 22-25, 2009, pp. 103-108.

[55] Richter, T., Boehringer, D. & Jeschke, S., 2009, "LiLa: A European project on networked experiments", *Proceedings of 6th International Conference on Remote Engineering and Virtual Instrumentation*, Bridgeport, Connecticut, USA, June 22-25, 2009.

[56] Machotka, J., Nedic, Z. & Gol, O., 2008, "Collaborative learning in the remote laboratory NetLab", *Journal on Systemics, Cybernetics and Informatics,* Vol. 6, No. 3, pp. 22-27.

[57] García-Zubía, J., López de Ipiña, D., Orduña, P., Hernández, U. & Trueba, I., 2006, "Evolution of the WebLab at the University of Deusto", *Proceedings of the 6th European Workshop on Microelectronics Education*, Stockholm, Sweden, June 8-9, 2006.

[58] Abdulwahed, M. & Nagy, Z. K., 2011, "The TriLab, a novel ICT based triple access mode laboratory education model", *Computers and Education*, Vol. 56 No. 1, pp. 262–274. http://dx.doi.org/10.1016/j.compedu.2010.07.023

[59] Gustavsson, I., 2003, "A remote access laboratory for electrical circuit experiments", *International Journal of Engineering Education*, Vol. 19, No. 3, pp. 409-419.

[60] Control labs online, University of Tennessee at Chattanooga, http://chem.engr.utc.edu/, accessed in August 2012.

[61] Schauer, F., Lustig, F., & Ozvoldova, M., 2006, "Remote scientific experiments across internet: Photovoltaic cell characterization", *Proceedings* of *International Conference on Interactive Collaborative Learning*, Villach, Austria, September 27- 29, 2006.

[62] Remotely controlled laboratory, Department of Experimental Physics, Olomouc, http://ictphysics.upol.cz/remotelab/index_en.html, accessed in August 2012.

[63] The telelabs project, School of Mechanical Engineering, The University of Western Australia, http://telerobot.mech.uwa.edu.au/, accessed in August 2012.

[64] Remote internet laboratory, GymKT, http://remote-lab.fyzika.net/, accessed in August 2012.

[65] Ma, J. & Nickerson, J. V., 2006, "Hands-on, simulated and remote laboratories: a comparative literature review", *ACM Computing Surveys*, Vol. 38, No. 3, pp. 1-24. http://dx.doi.org/10.1145/1132960.1132961

[66] Corter, J. E., Esche, S. K., Chassapis, C., Ma, J. & Nickerson, J. V., 2011, "Process and learning outcomes from remotely operated, simulated, and hands-on student laboratories", *Computers and Education*, Vol. 57, No. 3, pp. 2054-2067. http://dx.doi.org/10.1016/j.compedu.2011.04.009

[67] Corter, J. E., Nickerson, J. V., Esche, S. K., Chassapis, C., Im, S. & Ma, J., 2007, "Constructing reality: a study of remote, hands-on and simulated laboratories", ACM Transactions on Computer-Human Interaction, Vol. 14, No. 2, Article 7, 2007.

[68] Tsuda, S., Scott, D., Doyle, J. & Jones, D. B., 2009, "Surgical skills training and simulation", *Current Problems in Surgery*, Vol. 46, No. 4, pp. 271-370. http://dx.doi.org/10.1067/j.cpsurg.2008.12.003

[69] Prensky, M., 2001, "True believers: digital game-based learning in the military", *Chapter 10 of Digital Game-Based Learning*, McGraw-Hill, 2001.

[70] http://www.gamespot.com/school-tycoon/reviews/school-tycoon-review-6091764/, accessed in August 2012.

[71] Squire, K., 2005, "Changing the game: What happens when video games enter the classroom?", *Innovate: Journal of Online Education*, Vol. 1, No. 6.

[72] Burdea, G., 2003, "Virtual rehabilitation – Benefits and challenges", *Methods of Information in Medicine*, Vol. 42, No. 5, pp. 519-523.

[73] Coulouris, G., Dollimore, J. & Kindberg, T., 2005, "Distributed Systems: Concepts and Design", Ch. 18, 4nd Ed., Addison Wesley.

[74] Dai, S., Aziz, E.-S., Esche, S. K. & Chassapis, C., 2008, "A remotely accessed flow rig student laboratory", *Proceedings of the ASME International Mechanical Engineering Congress and Exposition* IMECE'08, Boston, Massachusetts, USA, October 31 - November 6, 2008.

[75] 3ds Max from Autodesk, Inc., http://usa.autodesk.com/3ds-max/, accessed in August 2012.

[76] Dassault Systemes SolidWorks Corp., http://www.solidworks.com/, accessed in August 2012.

[77] Havok physics engine, http://www.havok.com/products/physics, accessed in August 2012.

[78] Aziz, E.-S., Chang, Y., Esche, S. K. & Chassapis, C., 2012, "Capturing assembly constraints of experimental setups in a virtual laboratory environment". *Proceedings of the ASME International Mechanical Engineering Conference & Exposition* IMECE'12, Houston, Texas, USA, November 9-15, 2012.

[79] Frederick, B. P., 1999, "What's real about virtual reality?", *IEEE Computer Graphics and Applications*, Vol. 19, pp. 16-27. http://dx.doi.org/10.1109/38.799723

[80] Boian, R., Sharma, A., Han, C., Merians, A., Burdea, G., Adamovich, S., Recce, M., Tremaine, M. & Poizner, H., 2002, "Virtual reality-based post-stroke hand rehabilitation", *Proceedings of Medicine Meets Virtual Reality 2002 Conference*, Newport Beach, CA, pp. 64-70.

[81] Correa, A. G. D.; de Assis, G. A.; do Nascimento, M.; Ficheman, I. & de Deus Lopes, R., 2007, "GenVirtual: an augmented reality musical game for cognitive and motor rehabilitation", *Proceeding of Virtual Rehabilitation,* 2007, Venice, Italy, pp. 1-6.

[82] Sisto, S. A., Forrest, G. F. & Glendinning, D., 2002, "Virtual reality applications for motor rehabilitation after stroke", *Proceeding of Topics in Stroke Rehabilitation*, Vol. 8, No. 4., pp. 11-23. http://dx.doi.org/10.1310/YABD-14KA-159P-MN6F

[83] SeeMe rehabilitation system, http://www.virtual-reality-rehabilitati on.com/products/seeme/what-is-seeme, accessed in August 2012.

[84] Mihalik, J. P., Kohli, L. & Whitton, M. C., 2008, "Do the physical characteristics of a virtual reality device contraindicate its use for balance assessment?", *Journal of Sport Rehabilitation*, Vol. 16, pp. 38-49.

[85] Bailenson, J. N., Patel , K., Nielsen, A., Bajcsy, R., Jung, S. & Kurillo, G., 2008, "The effect of interactivity on learning physical actions in virtual reality", *Media Psychology*, Vol. 11, pp. 354–376. http://dx.doi.org/10.1080/15213260802285214

[86] Wilson, C., 2008, "Avatars, virtual reality technology and the U.S. military: emerging policy issues", *CRS Report for Congress*, Order Code RS22857.

[87] "Virtual reality in military", available on the Web, http://www.vrs.org.uk/virtual-reality-military/index.html, accessed in August 2012.

[88] Marques, A., Queirós, C. & Rocha, N., 2008, "Virtual reality and neuropsychology: a cognitive rehabilitation approach for people with psychiatric disabilities", *Proceeding of the Seventh International Conference on Disability, Virtual Reality and Associated Technologies with ArtAbilitation 2008*, Maia & Porto, Portugal, Session I, pp. 39-46.

[89] Peñate, W., Pitti, C. T., Bethencourt, J. M., de la Fuente, J. & Gracia, R., 2008, "The effects of a treatment based on the use of virtual reality exposure and cognitive-behavioral therapy applied to patients with agoraphobia", *International Journal of Clinical and Health Psychology*, Vol. 8, No. 1, pp. 5-22.

[90] Botella, C., Baños, R. M., Guerrero, B., García Palacios, A., Quero, S. & Alcañíz, M., 2006, "Using a flexible virtual environment for treating a storm phobia", *Psychology Journal*, Vol. 4, No. 2, pp. 129-144.

[91] Huang M. P. & Alessi N. E., 1998, "Current limitations into the application of virtual reality to mental health research", *ISO Press, 1998, Amsterdam, Netherlands*

[92] Freeman, D., 2008, "Studying and treating schizophrenia using virtual reality: a new paradigm", *Schizophrenia Bulletin*, Vol. 34, No. 4, pp. 605-610. http://dx.doi.org/10.1093/schbul/sbn020

[93] Rizzo, A. A., Schultheis, M. T. & Rothbaum, B. O., 2002, "Ethical issues for the use of virtual reality in the psychological sciences", In: S. Bush & M. Drexler (Eds.), "*Ethical Issues in Clinical Neuropsychology*", Swets & Zeitlinger Publishers, Lisse, Netherlands, pp. 243-280.

[94] Kijima, R., Shirakawa, K., Hirose, M. & Nihei, K., 1994, "Virtual sand box: development of an application of virtual environments for clinical medicine", *The MIT Press*, Vol. 3, pp. 45-59.

[95] Davis, A., "What about the digital toys? Looking into the idea of using digital media in play therapy sessions", http://www.mlppubs online.com/display_article.php?id=887087, accessed in August 2012.

AUTHORS

Zhou Zhang. Ph.D. candidate, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email: ZZhang11@stevens.edu).

Mingshao Zhang, Ph.D. candidate, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email: MZhang3@stevens.edu).

Serdar Tumkor, Research Scientist, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email: STumkor@stevens.edu) .

Yizhe Chang, Ph.D. candidate, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email:YChang1@stevens.edu).

Sven K. Esche, Associate Professor, Graduate Program Director, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email: SEsche@stevens.edu).

Constantin Chassapis, Professor, Vice Provost for Academics, Mechanical Engineering Department, Stevens Institute of Technology, Hoboken, NJ 07030 USA. (Email: CChassap@stevens.edu).

Submitted 23 April 2013. Published as re-submitted by the authors 15 Septemeber2013.